UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/821,687 | 04/09/2004 | F. Soner Terek | MSFT-2955/307064.01 | 1355 |

41505 7590 03/21/2008

WOODCOCK WASHBURN LLP (MICROSOFT CORPORATION)
CIRA CENTRE, 12TH FLOOR
2929 ARCH STREET
PHILADELPHIA, PA 19104-2891

| EXAMINER | |
|---|---|
| COLAN, GIOVANNA B | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2162 | |

| MAIL DATE | DELIVERY MODE |
|---|---|
| 03/21/2008 | PAPER |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 10/821,687 | TEREK ET AL. | |
| | Examiner | Art Unit | |
| | GIOVANNA COLAN | 2162 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *19 December 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *38, 40, 41 – 49, 54-55, and 57 – 58* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *38, 40, 41 – 49, 54-55, and 57 – 58* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

## DETAILED ACTION

1. This action is issued in response to the Amendment filed on 12/19/2007.

2. No claims were amended. Claims 1 – 37, 39, 50 – 53, and 56 were canceled. No claims were added.

4. Claims 38, 40, 41 – 49, 54-55, and 57 – 58 are pending in this application.

### *Response to Arguments*

5. Applicant's arguments filed on 12/19/2007have been fully considered but they are not persuasive.

6. This action is made final.

### *35 USC § 101*

7. For examination purposes, claims 38, 40, 41 – 49, 54-55, and 57 – 58, the term "computer readable medium bearing a computer readable representation of an object…., and wherein serialized for retrieval by computer hardware…" has been interpreted as tangible media, such as floppy diskettes, CD-ROMs, and hard drives (Also, see [0187], lines 7 – 11, Specification of the disclosure).

### *Claim Rejections - 35 USC § 103*

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

9. Claims 38, and 44 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bennion (U.S. Patent No. 5,634,123) in view of Hong et al. (Hong hereinafter) (US 6,266,673 B1).

Regarding claim 38, Bennion teaches a computer readable medium bearing a computer readable representation ... comprising:

a binary fragment (Fig. 2, item 201, Col. 7, lines 7 – 10, Bennion) associated with said object, said binary fragment comprising a binary fragment header and a binary fragment payload (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type", wherein a code

point is equivalent to a header and one of the types is data-containing records, in which case the header is a binary fragment header; Figure 2, element 205, wherein the Data portion is equivalent to the payload, Bennion[1]),

Bennion does not explicitly teach a binary fragment payload wherein said plurality of primitive data members are all of the primitive data members of the object. However, since each record within the COMPANY object may be classified as either a container record or a data-containing record (i.e. a primitive data member) (col. 3, lines 19-22, wherein the "Company" (Figure 6, item 603, Bennion) corresponds to the object claimed, Bennion) and also since "COMPANY DATA 601" is a data-containing record that contains data only (not other records) and also since it "describes the company as a whole", then it is suggested that data-containing record "COMPANY DATA 601" must contain all the primitive data members of object "COMPANY" (Col. 3, lines 5 – 8, Bennion). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to include all of the primitive data members of the object in a binary fragment payload as taught by Bennion to provide a complete COMPANY object record as shown in the example given in col. 3 lines 19-31. One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

Furthermore, Bennion discloses:

wherein the binary fragment header comprises a type field (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-

[1] Wherein the "Company" (Figure 6, item 603, Bennion) corresponds to the object claimed.

containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type.", wherein the data-containing record type is equivalent to a binary fragment type) and a length field (Figure 2.; col. 1, lines 60-62, "A length field facilitates variable data lengths for data-containing records", Bennion);

wherein said primitive data members comprise only members of a primitive data type and excluding at least collections (Fig. 6, item 601, Col. 3, lines 19 – 27, "...Container records contain other records, while data-containing records contain data. No record is both container record and a data-containing record..."; wherein the Examiner interprets that since item 601 "COMPANY DATA", Fig. 6, does not contain other records, as opposed to "REGION 602", then it exclude at least collections as claimed, Bennion[2]).

However, Bennion does not explicitly disclose that said primitive data type comprises at least integers. On the other hand, Hong discloses primitive data type comprising at least integers (Col. 7, lines 13 – 19, Hong). It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the Hong's teachings to the system of Bennion. Skilled artisan would have been motivated to do so, as suggested by Hong (Col. 7, lines 1 – 4 and 13 – 19, Hong), to be able to

[2] Examiner makes note that the specification provides examples of "primitive members", (specification of the disclosure; [0006], "string", "integer") also as defined by the dictionary ("Academic Press Dictionary of Science and Technology from Elsevier Science & Technoloy, Copyright 1992, 1996 by Academic Press") Primitive is a fundamental unit that cannot be divided. Therefore, data (in a data-containing record which does NOT contain other records) is of a primitive data type. See also, figure 3, Bennion, first block of data starting from the left on "151-200", the payload "DATA" includes "123 Avenue "E" " which are string and integers. Examiner has interpreted the claims in light of the specification. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

define object types, object tables, or relational tables, in this case: primitive data type INTEGER.

The combination of Bennion in view of Hong also discloses:

at least one additional fragment comprising at least one non-primitive member of the object (See at least col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type"; Figure 3, and col. 6, lines 5-18, wherein "a sample string of data bytes representing a series of hierarchically-organized records" illustrates a fragment comprising non-primitive members of the object, Bennion).

Furthermore, the combination of Bennion in view of Hong discloses:

wherein the primitive data members are in a storage engine record format (col. 3, lines 19-24, "Thus, a hierarchical structure emerges. Each individual record contained within COMPANY record 600 may be classified as either a container record or a data-containing record. Container records contain other records, while data-containing records contain data. No record is both a container record and a data-containing record"; col. 4, lines 40-43, "By definition, a data-containing record has a "Length" equal to the actual length of the record itself, since a data-containing record cannot contain another record", wherein a data-containing record is a primitive data member, Bennion; and also see Col. 3, lines 55 – 66, **"Each record in table corresponds to one record..", see for example row: "7-33", "Company Data", "27"**; Bennion; and

also see Col. 7, lines 13 – 19, "...The above statement defines the **relational table as having a column called NAME and specifies the column's data type as a VARCHAR** having a maximum length of 30. **VARCHAR is a primitive data type recognized by a DBMS, such as Oracle8.TM.. A VARCHAR is a string that is variable in length. Other examples Of primitive data type include INTEGER and FLOAT**..."; Hong).

Claim 44 is rejected for the reasons set forth hereinabove for claim 38 and furthermore the combination of Bennion in view of Hong discloses a medium comprising a terminator fragment that marks the end of the object, said terminator fragment comprising a terminator type field indicating the terminator fragment is a terminator fragment (col. 6, lines 40-42, Bennion).

10. Claims 40-43 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bennion (U.S. Patent No. 5,634,123), in view of Hong et al. (Hong hereinafter) (US 6,266,673 B1) further in view of Krishnaprasad et al, "Krishnaprasad" (U.S. Publication No. 2004/0220946), and Sarkar (U.S. Patent No. 6,012,067).

Claim 40 is rejected for the reasons set forth hereinabove for claim 38. However, the combination of Bennion in view of Hong does not explicitly disclose a fragment comprising Large Object (LOB) fragment.

Krishnaprasad discloses a computer readable medium comprising:

at least one Large Object (LOB) fragment comprising a LOB fragment header (page 5, section [0062], "Message 300 includes four fields: a length field 302; a version field 304; a flag field 306; and a payload field 310. The combination of fields 302, 304, 306, 310 composes a serialized image of XML data, according to the illustrated embodiment", wherein the fields except the payload field are considered to be the header portion) and a LOB fragment payload (page 6, section [0066], "Payload field 310 includes serialized XML data for a particular XML construct");

wherein the LOB header comprises a LOB type field, wherein the LOB type field indicates the LOB fragment is a LOB fragment (page 6, section [0066], "If the flag field 306 indicates that the type is LOB, then a locator for the LOB appears in the payload, such as the octal string "0.times.000030303030303- 0 . . . ");

and a LOB length field, wherein the LOB length field indicates a length of the LOB fragment payload (page 5, section [0063], "Length field 302 includes data that indicates the length of the serialized image. Any method known in the art for indicating length may be used."; Page 6, TABLE 2).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate a LOB fragment structure as disclosed by Krishnaprasad into the computer readable representation as disclosed in the combination of Bennion in view of Hong so that an XML element may be transferred between processes 132a and 132b, which both have access to LOB 144b, by passing the LOB locator (page 4, section [0047]). One of ordinary skill in the art would be

motivated to make the aforementioned combination with reasonable expectation of success.

However, the combination of Bennion in view of Hong and Krishnaprasad does not explicitly disclose a value type field that indicates whether the LOB fragment payload comprises an inline LOB or a pointer to a LOB location.

Sarkar teaches a value type field, wherein the value type field indicates whether the LOB fragment payload comprises an inline LOB or a pointer to a LOB location (col. 4, lines 25-33, "Internal LOB columns contain LOB locators that can refer to out-of-line or inline LOB values. Selecting a LOB column value returns the LOB locator and not the entire LOB value. Different operations in the form of packages and functions are performed through these locators. Multiple LOB data type columns can be defined in a table and all possible SQL operations are possible over such tables and attributes. LOB locator can be stored in the table column, either with or without the actual LOB value").

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate a value type field, as disclosed by Sarkar, into the computer readable representation as disclosed in the combination of Bennion in view of Hong and further in view of Krishnaprasad so that when a LOB column value is selected, the LOB locator is first returned instead of the entire LOB value (col. 4, lines 27-28). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

Claim 41 is rejected for the reasons set forth hereinabove for claim 40 and furthermore the combination of Bennion in view of Hong in view of Krishnaprasad and further in view of Sarkar discloses a medium wherein the LOB fragment payload comprises a LOB (col. 4, lines 25-33, Sarkar).

Claim 42 is rejected for the reasons set forth hereinabove for claim 40 and furthermore the combination of Bennion in view of Hong in view of Krishnaprasad and further in view of Sarkar discloses a medium wherein the LOB fragment payload comprises a pointer to a LOB location (col. 4, lines 25-33, Sarkar).

Claim 43 is rejected for the reasons set forth hereinabove for claim 40 and furthermore the combination of Bennion in view of Hong in view of Krishnaprasad and further in view of Sarkar discloses a medium wherein the value type field indicates whether the LOB fragment payload comprises an inline LOB, a pointer to a LOB location, or a cell reference (col. 4, lines 25-33; col. 5, lines 18-47, Sarkar).

11. Claims 45-48 and 55-57 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bennion (U.S. Patent No. 5,634,123), in view of Hong et al. (Hong hereinafter) (US 6,266,673 B1), and further in view of Stickler (US Patent No. 6,904,454 B2).

Claim 45 is rejected for the reasons set forth hereinabove for claim 38 and furthermore the combination of Bennion in view of Hong discloses a medium wherein said at least one additional fragment comprises:

a collection start fragment comprising a collection start header (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. **A code point found at the beginning of each record specifies its type**", wherein the fragment indicating the code element corresponds to the collection start fragment as claimed; also Examiner makes note that Bennion's code point is at the beginning of each record; therefore, the code is a collection start fragment as claimed; Bennion[3]);

wherein the collection start header comprises a collection start type field, wherein the collection start type field indicates the collection start fragment is a collection start fragment (Col. 1, lines 56 - 63, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. **A code point found at the beginning of each record specifies its type**. A length field facilitates variable data lengths for data-containing records, as well as implicit definition of a hierarchical structure among records", wherein container records are equivalent to "collection". "... a hierarchical structure among records..." indicates the structure of a

[3] Also see, Col. 6, lines 12 – 16, Bennion, "In the code points shown, the first byte is CO for container records ...". As also shown in Figure 3, first row 1 – 50, Bennion, the "CP" (code point) is always at the start of the blocks of data. For example, first block CP=CONT with LEN=302, second block CP=CONT with LEN=98, third block CP...etc.

collection type record; col. 5, lines 18-20, "**One bit is used to indicate that this is a container record** (as opposed to a data-containing record)") indicate that this is a container record (as opposed to a data-containing record)", Bennion[4]); and

a plurality of collection element fragments (See at least col. 1, lines 56 – 63; Figure 3, and col. 6, lines 5-18).

However, the combination of Bennion in view of Hong does not explicitly disclose a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments.

Stickler, on the other hand, discloses a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments (Col. 1 and 17, lines 56 – 58 and 45 – 47; respectively, "at least one entity includes metadata that identifies a sequential relationship between one or more entities within the scope of said one entity, each of said entities including metadata defining a position within said sequential relationship", wherein the one entity that identifies a sequential relationship (i.e. order of entities) corresponds to the bit field that indicates whether an order exists among collection element fragments as claimed, Stickler).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to modify the combined teachings of Bennion and Hong by incorporating a bit field indicating whether an order exists as disclosed by Stickler

[4] Also see, Col. 6, lines 12 – 16, Bennion, "In the code points shown, the first byte is CO for container records ...". As also shown in Figure 3, first row 1 – 50, Bennion, the "CP" (code point) is always at the start of the blocks of data. For example, first block CP=CONT with LEN=302, second block CP=CONT with LEN=98, third block CP...etc.

among a plurality of collection element fragments, to overcome a difficult situation present in known tree-based versioning models namely their inability to explicitly define relationships between different releases (col. 2, lines 1-5). Furthermore, MARS 25 also provides encoding properties defining special qualities relating to the format, structure or general serialization of data streams (col. 10, lines 16-18). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

Claim 46 is rejected for the reasons set forth hereinabove for claim 45 and furthermore the combination of Bennion in view of Hong and further in view of Stickler discloses a medium comprising:

at least one collection element fragment comprising a collection element header and collection element payload (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type"; col. 5, lines 18-20, "One bit is used to indicate that this is a container record (as opposed to a data-containing record)", wherein a code point is equivalent to a header and one of the types is container records, in which case the header is a collection element header because container records contain other records, which is equivalent to a "collection" element, and the data itself is the payload, Bennion);

wherein the collection element header comprises a collection element type field, wherein the collection element type field indicates the collection element fragment is a collection element fragment; (col. 5, lines 18-20, "One bit is used to indicate that this is a container record (as opposed to a data-containing record)", Bennion),

and a collection element length field, wherein the collection element length field indicates the a length of the collection element payload (col. 5, lines 23-25, "For container records, the length field 203 specifies the total length of all records that the current record contains (as described above)", Bennion);

Claim 47 is rejected for the reasons set forth hereinabove for claim 46, and furthermore the combination of Bennion in view of Hong and further in view of Stickler discloses a medium wherein the collection element payload comprises a data member in a collection of data members corresponding to said collection start fragment (col. 3, lines 32-43, Bennion).

Claim 48 is rejected for the reasons set forth hereinabove for claim 46, and furthermore the combination of Bennion in view of Hong and further in view of Stickler discloses a medium wherein

the collection element header further comprises a collection element locator field that provides a unique location of a data member in a collection of data members (col. 4, lines 24-31, Bennion).

Claim 55 is rejected on grounds corresponding to the reasons given above for claims 45 and 46.

Claims 56, 57 are rejected on grounds corresponding to the reasons given above for claims 47 and 48.

12. Claims 49 and 54 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bennion (U.S. Patent No. 5,634,123), in view of Krishnaprasad et al, "Krishnaprasad" (U.S. Publication No. 2004/0220946), Sarkar (U.S. Patent No. 6,012,067), and Stickler (US Patent No. 6,904,454 B2).

With respect to claim 49, Bennion teaches a computer readable medium bearing a computer readable representation ... comprising:

a collection start fragment comprising a collection start header (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type", wherein container records are equivalent to "collection". A code point is equivalent to the start header, Bennion);

wherein the collection start header comprises a collection start type field, wherein the collection start type field indicates the collection start fragment is a collection start

fragment (col. 1, lines 56 - 63, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type. A length field facilitates variable data lengths for data-containing records, as well as implicit definition of a hierarchical structure among records", wherein container records are equivalent to "collection". A code point is equivalent to the start header. "a hierarchical structure among records" indicates the structure of a collection type record; col. 5, lines 18-20, "One bit is used to indicate that this is a container record (as opposed to a data-containing record)", Bennion); and

a plurality of collection element fragments (See at least col. 1, lines 56 – 63; Figure 3, and col. 6, lines 5-18, Bennion) associated with said collection start fragment (Fig. 3, Col. 6, lines 5 – 15, "...the first byte is CO for container records...", Bennion), each of said collection element fragments comprising a collection element header and a collection element payload, wherein each collection payload comprises only a data member of a collection element data type (Col. 5 and 6, lines 5 – 13 and 9 – 15; respectively, Bennion), said collection element data type comprising data of same type as every collection element associated with said collection start fragment (Col. 3 and 6, lines 19 – 24 and 9 – 16; "No record is both a container record and a data-containing record..." and "...and 'DATA' is data field. In the code points shown, the first byte is CO for container records..."; respectively, wherein the first byte CO indicates that the data is of same type, such as, container record, Bennion).

However, Bennion does not explicitly disclose a fragment comprising Large Object (LOB) fragment and nor does Bennion explicitly disclose a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments.

Krishnaprasad discloses a computer readable medium bearing ... representation ... comprising:

at least one Large Object (LOB) fragment comprising a LOB fragment header (page 5, section [0062], "Message 300 includes four fields: a length field 302; a version field 304; a flag field 306; and a payload field 310. The combination of fields 302, 304, 306, 310 composes a serialized image of XML data, according to the illustrated embodiment", wherein the fields except the payload field are considered to be the header portion) and a LOB fragment payload (page 6, section [0066], "Payload field 310 includes serialized XML data for a particular XML construct", Krishnaprasad);

wherein the LOB header comprises a LOB type field, wherein the LOB type field indicates the LOB fragment is a LOB fragment (page 6, section [0066], "If the flag field 306 indicates that the type is LOB, then a locator for the LOB appears in the payload, such as the octal string "0.times.000030303030303- 0 . . . ", Krishnaprasad);

and a LOB length field, wherein the LOB length field indicates a length of the LOB fragment payload (page 5, section [0063], "Length field 302 includes data that indicates the length of the serialized image. Any method known in the art for indicating length may be used."; Page 6, TABLE 2, Krishnaprasad).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate a LOB fragment structure as disclosed by Krishnaprasad into the computer readable representation as disclosed in Bennion so that an XML element may be transferred between processes 132a and 132b, which both have access to LOB 144b, by passing the LOB locator (page 4, section [0047]). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

However, the combination of Bennion in view of Krishnaprasad does not explicitly disclose a value type field that indicates whether the LOB fragment payload comprises an inline LOB or a pointer to a LOB location, nor a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments

Sarkar teaches a value type field, wherein the value type field indicates whether the LOB fragment payload comprises an inline LOB or a pointer to a LOB location (col. 4, lines 25-33, "Internal LOB columns contain LOB locators that can refer to out-of-line or inline LOB values. Selecting a LOB column value returns the LOB locator and not the entire LOB value. Different operations in the form of packages and functions are performed through these locators. Multiple LOB data type columns can be defined in a table and all possible SQL operations are possible over such tables and attributes. LOB locator can be stored in the table column, either with or without the actual LOB value", Sarkar).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate a value type field, as disclosed by Sarkar, into the

computer readable representation as disclosed in the combination of Bennionin view of Krishnaprasad so that when a LOB column value is selected, the LOB locator is first returned instead of the entire LOB value (col. 4, lines 27-28, Sarkar). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

However, the combination of Bennion, Krishnaprasad and Sarkar does not explicitly disclose a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments.

Stickler discloses a bit field, wherein the bit field indicates whether an order exists among a plurality of collection element fragments (col. 1, lines 56-58, "at least one entity includes metadata that identifies a sequential relationship between one or more entities within the scope of said one entity, each of said entities including metadata defining a position within said sequential relationship", wherein the one entity that identifies a sequential relationship (i.e. order of entities) is analogous to the bit field that indicates whether an order exists among collection element fragments as claimed, Stickler).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate a bit field indicating whether an order exists as disclosed by Stickler among a plurality of collection element fragments, as disclosed in Bennion, to overcome a difficult situation present in known tree-based versioning models namely their inability to explicitly define relationships between different releases (col. 2, lines 1-5, Stickler). Furthermore, MARS 25 also provides encoding properties

defining special qualities relating to the format, structure or general serialization of data streams (col. 10, lines 16-18). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

Claim 54 is rejected for the reasons set forth hereinabove for claim 53 and furthermore Bennion, in view of Krishnaprasad, in view of Sarkar, and further in view of Stickler discloses a medium comprising:

a collection element fragment comprising a collection element header and collection element payload (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type"; col. 5, lines 18-20, "One bit is used to indicate that this is a container record (as opposed to a data-containing record)", wherein a code point is equivalent to a header and one of the types is container records, in which case the header is a collection element header because container records contain other records, which is equivalent to a "collection" element, and the data itself is the payload, Bennion);

wherein the collection element header comprises a collection element type field, wherein the collection element type field indicates the collection element fragment is a collection element fragment; (col. 5, lines 18-20, "One bit is used to indicate that this is a container record (as opposed to a data-containing record)", Bennion),

and a collection element length field, wherein the collection element length field indicates a length of the collection element payload (col. 5, lines 23-25, "For container records, the length field 203 specifies the total length of all records that the current record contains (as described above)", Bennion);

13. Claim 58 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bennion (U.S. Patent No. 5,634,123) in view of Hong et al. (Hong hereinafter) (US 6,266,673 B1) and Roy et al. ("Roy" hereinafter) (US Patent No. 6,631,130).

Regarding claim 58, Bennion teaches a computer readable medium bearing a computer readable representation ... comprising:

a binary fragment (Fig. 2, item 201, Col. 7, lines 7 – 10, Bennion) associated with said object, said binary fragment comprising a binary fragment header and a binary fragment payload (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type", wherein a code point is equivalent to a header and one of the types is data-containing records, in which case the header is a binary fragment header; Figure 2, element 205, wherein the Data portion is equivalent to the payload, Bennion[5]),

Bennion does not explicitly teach a binary fragment payload wherein said plurality of primitive data members are all of the primitive data members of the object.

[5] Wherein the "Company" (Figure 6, item 603, Bennion) corresponds to the object claimed.

However, since each record within the COMPANY object may be classified as either a container record or a data-containing record (i.e. a primitive data member) (col. 3, lines 19-22, wherein the "Company" (Figure 6, item 603, Bennion) corresponds to the object claimed, Bennion) and also since "COMPANY DATA 601" is a data-containing record that contains data only (not other records) and also since it "describes the company as a whole", then it is suggested that data-containing record "COMPANY DATA 601" must contain all the primitive data members of object "COMPANY" (Col. 3, lines 5 – 8, Bennion). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to include all of the primitive data members of the object in a binary fragment payload as taught by Bennion to provide a complete COMPANY object record as shown in the example given in col. 3 lines 19-31. One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

Furthermore, Bennion discloses:

wherein the binary fragment header comprises a type field (col. 1, lines 56-60, "The data structure defined by the present invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type.", wherein the data-containing record type is equivalent to a binary fragment type) and a length field (Figure 2.; col. 1, lines 60-62, "A length field facilitates variable data lengths for data-containing records", Bennion);

wherein said primitive data members comprise only members of a primitive data type and excluding at least collections (Fig. 6, item 601, Col. 3, lines 19 – 27, "…Container records contain other records, while data-containing records contain data. No record is both container record and a data-containing record…"; wherein the Examiner interprets that since item 601 "COMPANY DATA", Fig. 6, does not contain other records, as opposed to "REGION 602", then it exclude at least collections as claimed, Bennion[6]).

However, Bennion does not explicitly disclose that said primitive data type comprises at least integers. On the other hand, Hong discloses primitive data type comprising at least integers (Col. 7, lines 13 – 19, Hong). It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the Hong's teachings to the system of Bennion. Skilled artisan would have been motivated to do so, as suggested by Hong (Col. 7, lines 1 – 4 and 13 – 19, Hong), to be able to define object types, object tables, or relational tables, in this case: primitive data type INTEGER.

The combination of Bennion in view of Hong also discloses:

at least one additional fragment comprising at least one non-primitive member of the object (See at least col. 1, lines 56-60, "The data structure defined by the present

[6] Examiner makes note that the specification provides examples of "primitive members", (specification of the disclosure; [0006], "string", "integer") also as defined by the dictionary ("Academic Press Dictionary of Science and Technology from Elsevier Science & Technoloy, Copyright 1992, 1996 by Academic Press") Primitive is a fundamental unit that cannot be divided. Therefore, data (in a data-containing record which does NOT contain other records) is of a primitive data type. See also, figure 3, Bennion, first block of data starting from the left on "151-200", the payload "DATA" includes "123 Avenue "E" " which are string and integers. Examiner has interpreted the claims in light of the specification. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

invention includes two types of records: data-containing records and container records. Data-containing records contain data, while container records contain other records. A code point found at the beginning of each record specifies its type"; Figure 3, and col. 6, lines 5-18, wherein "a sample string of data bytes representing a series of hierarchically-organized records" illustrates a fragment comprising non-primitive members of the object, Bennion).

Furthermore, the combination of Bennion in view of Hong discloses:

wherein the primitive data members are in a storage engine record format (col. 3, lines 19-24, "Thus, a hierarchical structure emerges. Each individual record contained within COMPANY record 600 may be classified as either a container record or a data-containing record. Container records contain other records, while data-containing records contain data. No record is both a container record and a data-containing record"; col. 4, lines 40-43, "By definition, a data-containing record has a "Length" equal to the actual length of the record itself, since a data-containing record cannot contain another record", wherein a data-containing record is a primitive data member, Bennion; and also see Col. 3, lines 55 – 66, **"Each record in table corresponds to one record..", see for example row: "7-33", "Company Data", "27"**; Bennion; and also see Col. 7, lines 13 – 19, "...The above statement defines the **relational table as having a column called NAME and specifies the column's data type as a VARCHAR** having a maximum length of 30. **VARCHAR is a primitive data type recognized by a DBMS, such as Oracle8.TM.. A VARCHAR is a string that is**

**variable in length. Other examples Of primitive data type include INTEGER and FLOAT**..."; Hong).

Neither Bennion nor Hong discloses the claimed "type field indicates that the binary fragment is the only fragment of the object".

On the other hands, Roy discloses that the type field indicates that the binary fragment is the only fragment of the object (col.3, lines 52-59, "The remaining sixty-eight bits of the PDU are used for various other addressing information such as indicating whether the PDU contains an ATM cell, a packet, or a control message, whether reassembly of the packet should be aborted, whether the payload is a first fragment, middle fragment or last fragment, how many payload bytes are in the last fragment, the fragment sequence count, and a destination flow identifier", wherein the fragment sequence count is used to show the fragment count, Roy).

It would have been obvious to one having ordinary skill in the art at the time the invention was made to incorporate the type field indicating that the binary fragment is the only fragment of the object as disclosed by Roy into the binary fragment data representation as disclosed in Bennion for part of addressing information (col. 3, lines 52-54). One of ordinary skill in the art would be motivated to make the aforementioned combination with reasonable expectation of success.

***Response to Arguments***

1. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on

combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

2. Applicant argues that; "it appears the Official Action considers Bennion's data-containing record equivalent to a primitive data member as recited in Applicants' claims. This is incorrect. Bennion makes no mention that "data containing records" are equivalent to primitive data members, as the term is understood by persons of skill in the art".

Examiner respectfully disagrees. The combination of Bennion in view of Hong does disclose the primitive data members claimed (Fig. 6, item 601, Col. 3, lines 19 – 27, "…Container records contain other records, while data-containing records contain data. No record is both container record and a data-containing record…", wherein data-containing record "COMPANY DATA" 601 corresponds to the primitive data members claimed; Bennion). Also, the examiner makes note that the specification provides examples of "primitive members", (specification of the disclosure; [0006], "string", "integer") also as defined by the dictionary ("Academic Press Dictionary of Science and Technology from Elsevier Science & Technoloy, Copyright 1992, 1996 by Academic Press") Primitive is a fundamental unit that cannot be divided. Therefore, data (in a data-containing record which does NOT contain other records; therefore, cannot be divided by other records) is of a primitive data type. See also, figure 3, Bennion, third block of data starting from the left on "1 -50", the payload "DATA" includes "John Doe" (which is of type string). Examiner has interpreted the claims in light of the specification. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

3. Applicant argues that; "it appears that the Office Action considers Applicants' claimed binary fragment to be equivalent to the COMPANY container record disclosed in Bennion..." and further argues that; "The problem is that the Official Action is first asserting that Applicants' claimed binary fragment is equivalent to Bennion's COMPANY container record, and therefore contains all the records for the entire object , and next asserting that Bennion discloses at least one additional fragment...".

Examiner respectfully disagrees. First, "COMPANY" record (in Figure 6, Bennion) corresponds to the **"object"** recited in Applicant's claimed invention. Second, the data-containing record "COMPANY DATA" 601 (figure 6, Bennion) corresponds to the binary fragment associated with the object comprising all primitive data members as claimed. The container record "REGION" 602 (figure 6, Bennion) corresponds to the "additional fragment comprising at least one non-primitive member as claimed. As disclosed by Bennion, , "…No record is both container record and a data-containing record…", (Fig. 6, Col. 3, lines 19 – 27, Bennion).

4. Applicant argues that; "Bennion does not disclose primitive data members in a storage engine record format... A hierarchical structure as provided in Bennion is not a storage engine record format. Refer for example to Applicants' Fig. 10, which illustrates columns and rows."

Examiner respectfully disagrees. The combination of Bennion in view of Hong does teach the limitation including wherein the primitive data members are in a storage engine record format (col. 3, lines 19-24, "Thus, a hierarchical structure emerges. Each individual record contained within COMPANY record 600 may be classified as either a container record or a data-

containing record. Container records contain other records, while data-containing records contain data. No record is both a container record and a data-containing record"; col. 4, lines 40-43, "By definition, a data-containing record has a "Length" equal to the actual length of the record itself, since a data-containing record cannot contain another record", wherein a data-containing record is a primitive data member, Bennion; and also see Col. 3, lines 55 – 66, "**Each record in table corresponds to one record..", see for example row: "7-33", "Company Data", "27**"; Bennion; and also see Col. 7, lines 13 – 19, "…The above statement defines the **relational table as having a column called NAME and specifies the column's data type as a VARCHAR** having a maximum length of 30. **VARCHAR is a primitive data type recognized by a DBMS, such as Oracle8.TM.. A VARCHAR is a string that is variable in length. Other examples Of primitive data type include INTEGER and FLOAT**…"; Hong).

5. Applicant argues that; "it appears that the Official Action alleges that 'a binary fragment' as taught by Applicants' claims is both a container record and a data-containing record, in violation of the teachings of Bennion itself".

Examiner respectfully disagrees. Bennion's data-containing record (shown in Figure 2, item 201, Bennion), corresponds to the binary fragment claimed.

6. Applicant argues that; "The Official Action apparently concludes that a collection start fragment is equivalent to a container record as taught by Bennion… However, Bennion states that, 'container records contain other records.' Bennion col. 3, lines 22 – 24. In contrast, there is nothing in Applicants' claims or specification to indicate that a collection start fragment would 'contain other records.'…".

Examiner respectfully disagrees. First, the code point found at the beginning of each record (col. 1, lines 56-60, Bennion) corresponds to the collection start fragment as claimed. Second, the claimed language does not specifically recite that such "collection start fragment" excludes or does not contain "other records". Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). Third, Examiner notes that the claimed language (claim 45) recites "a collection start fragment **comprising** a collection start header…." which is inclusive or open-ended and does not exclude additional, unrecited elements or method steps. **See MPEP 2111.02:**

The transitional phrases "comprising", "consisting essentially of" and "consisting of" define the scope of a claim with respect to what unrecited additional components or steps, if any,are excluded from the scope of the claim. The transitional term "comprising", which is synonymous with "including," "containing," or "characterized by," is inclusive or open-ended and does not exclude additional, unrecited elements or method steps. See, e.g., > Mars Inc. v. H.J. Heinz Co., 377 F.3d 1369, 1376, 71 USPQ2d 1837, 1843 (Fed. Cir. 2004) ("like the term comprising,' the terms containing' and mixture' are open-ended.").< Invitrogen Corp. v. Biocrest Mfg., L.P., 327 F.3d 1364, 1368, 66 USPQ2d 1631, 1634 (Fed. Cir. 2003) ("The transition comprising' in a method claim indicates that the claim is open-ended and allows for additional steps."); Genentech, Inc. v. Chiron Corp., 112 F.3d 495, 501, 42 USPQ2d 1608, 1613 (Fed. Cir. 1997) **("Comprising" is a term of art used in claim language which means that the named elements are essential, but other elements may be added and still form a construct within the scope of the claim.)**; Moleculon Research Corp. v. CBS, Inc., 793 F.2d 1261, 229 USPQ

805 (Fed. Cir. 1986); In re Baxter, 656 F.2d 679, 686, 210 USPQ 795, 803 (CCPA 1981); Ex parte Davis, 80 USPQ 448, 450 (Bd. App. 1948) ("comprising" leaves "the claim open for the inclusion of unspecified ingredients even in major amounts"). >In Gillette Co. v. Energizer Holdings Inc., 405 F.3d 1367, 1371-73, 74 USPQ2d 1586, 1589-91 (Fed. Cir. 2005), the court held that a claim to "a safety razor blade unit comprising a guard, a cap, and a group of first, second, and third blades" encompasses razors with more than three blades because the transitional phrase "comprising" in the preamble and the phrase "group of" are presumptively open- ended. "The word comprising' transitioning from the preamble to the body signals that the entire claim is presumptively open-ended." Id. In contrast, the court noted the phrase "group consisting of" is a closed term, which is often used in claim drafting to signal a "Markush group" that is by its nature closed. Id. The court also emphasized that reference to "first," "second," and "third" blades in the claim was not used to show a serial or numerical limitation but instead was used to distinguish or identify the various members of the group. Id.<

7. Applicant argues that; "…Namely, a collection start fragment is a fragment at the start of a collection, not a container of other records as taught by Bennion".

Examiner respectfully disagrees. "The code point found at the beginning of each record" (as disclosed by Bennion, col. 1, lines 56-60) corresponds to the collection start fragment as claimed. Also see, Col. 6, lines 12 – 16, Bennion, "In the code points shown, the first byte is CO for container records ...". As also shown in Figure 3, first row 1 – 50, Bennion, the "CP" (code point) is always at the start of the blocks of data. For example, first block CP=CONT with

LEN=302, second block CP=CONT with LEN=98, third block CP...etc. Therefore, the code point corresponds to the collection start fragment claimed.

8. Applicant argues that; "there is not teaching or suggestion in Bennion that a container record contains a collection…", and further argues that; "Bennion provides not indication that such collection data would be contained in a container record."

Examiner respectfully disagrees. The applied art Bennion does disclose the claimed limitation including: only a data member of a collection element data type (Col. 5 and 6, lines 5 – 13 and 9 – 15; respectively, Bennion), said collection element data type comprising data of same type as every collection element associated with said collection start fragment (Col. 3 and 6, lines 19 – 24 and 9 – 16; "No record is both a container record and a data-containing record…" and "…and 'DATA' is data field. In the code points shown, the first byte is CO for container records…"; respectively, wherein the first byte CO indicates that the data is of same type, such as, container record, Bennion). Also, examiner points out that, Bennion (figure 6, Bennion) clearly shows that; for example, container record "Region 602" contains collection "DISTRICT 605" which as taught by Bennion "each describe a district associated with the region. Each DISTRICT record 605 is further divided into a DISTRICT DATA record 606 and two STORE DATA records 607" (Col. 3, lines 7 – 18, Bennion). Therefore, the Bennion's container does contained collection data.

9. Applicant argues that; "...while Bennion, some records identify themselves as containing other records, or containing data, no records identify themselves as starting a collection or as a member of a collection, as recited in Applicant's claims".

Examiner respectfully disagrees. The applied art does teach records identify themselves as starting a collection ("The code point found at the beginning of each record", as disclosed by Bennion, col. 1, lines 56-60). Also see, Col. 6, lines 12 – 16, Bennion, "In the code points shown, the first byte is CO for container records ...". As also shown in Figure 3, first row 1 – 50, Bennion, the "CP" (code point) is always at the start of the blocks of data. For example, first block CP=CONT with LEN=302, second block CP=CONT with LEN=98, third block CP...etc.

## Conclusion

1. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

2. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

### *Points Of Contact*

Any inquiry concerning this communication or earlier communications from the examiner should be directed to GIOVANNA COLAN whose telephone number is (571)272-2752. The examiner can normally be reached on 8:30 am - 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Breene can be reached on (571) 272-4107. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Giovanna Colan
Examiner
Art Unit 2162
March 17, 2008

/J. M. C./

/John Breene/
Supervisory Patent Examiner, Art Unit 2162